What Is Claimed Is:

1. A system for determining the start of a match of a regular expression, comprising:

a special state table which contains start state entries and terminal state entries;

a plurality of start state registers for storing offset information indicative of the start of a match of the regular expression;

a deterministic finite state automaton (DFA) next state table which, given the current state and an input character, returns the next state, the DFA next state table including a settable indicator for any next state table entry which indicates whether to perform a lookup into the special state table; and

a compiler which loads values into the special state table based on the regular expression.

2. A system for determining the start of one or more patterns of characters in an input character string, the patterns being defined by at least one character of the input character string, the input character string being provided to the system, the system operating in a series of states, the series of states including at least one start state and at least one terminal state, the system comprising:

finite state automaton, the finite state automaton being responsive to each character of the input character string and selectively transitioning to a next state in response to each character;

an automaton memory having stored therein a state transition table and a special state table;

the special state table including special state information;

the special state information including start state entries and terminal state entries, the special state information having at least a first code to indicate whether the special state information is a start state entry or a terminal state entry, each start state entry including a start state register select code, each terminal state entry including a second code identifying the one or more particular patterns, and a start state register number code; and

a plurality of start state registers, each register of the plurality of start state registers being identifiable by the start state register number code and having stored therein information relating to the location in the input character string of the start of a particular pattern of the one or more patterns;

the state transition table including current state information corresponding to the current state of the finite state automaton, character information corresponding to the characters in the input character string, next state information relating to the next state to which the finite state automaton will transition in response to the current state information and the character information, and special state table information corresponding to the next state information and indicating whether the system should perform a lookup in the special state table.

3. A system as defined by Claim 2, which further comprises:

a compiler cooperatively linked to the automaton memory, the compiler generating the special state information in the special state table and the current state information, character information, next state information and special state table information in the state transition table.

4. A system as defined by Claim 2, wherein the information stored in each register of the plurality of start state registers is offset information which corresponds to the position of a character in the input character string which resulted in the next state being a start state.

5. A method of determining the start of a match of a regular expression using a system having a special state table, a plurality of start state registers and a deterministic finite state automaton next state table, the method comprising the steps of:

determining, from the regular expression, each start state and each terminal state of a match of the regular expression;

loading a start state entry into the special state table for each start state;

loading a terminal state entry into the special state table for each terminal state;

determining a next state from a current state and an input character from an input character string;

loading a current offset from the beginning of the input character string into the start state register when a start state is encountered; and

retrieving from the special state table the terminal state entry and retrieving the current offset from the start state register pertaining to the match of the regular expression when a terminal state is encountered.

6. A method for determining the start of one or more patterns of characters in an input character string, the patterns being defined by at least one character of the input character string, the input character string being provided to a system having a finite state automaton, an automaton memory operatively linked to the finite state automaton, and a plurality of start state registers operatively linked to the automaton memory and finite state automaton, the system operating in a series of states, the series of states including at least one start state and at least one terminal state, the method comprising the steps of:

providing each character of the input character string to the system such that the finite state automaton is responsive thereto and selectively transitions from a current state to a next state in response to each character;

storing in the automaton memory a state transition table and a special state table, the special state table including special state information, the special state information including start state entries and terminal state entries, the special state information having at least a first code to indicate whether the special state information is a start state entry or a terminal state entry, the state transition table including current state information corresponding to the current state of the finite state automaton, character information corresponding to the characters in the input character string, next state information relating to the next state to which the finite state automaton will transition in response to the current state information and the character information, and special state table information corresponding to the next state information and indicating whether the system should perform a lookup in the special state table;

storing in each register of the plurality of start state registers information relating to the location in the input character string of the start of a particular pattern of the one or more patterns;

determining from the state transition table whether the next state is a special state in response to an input character of the input character string;

performing a lookup in the special state table if the next state is determined to be a special state;

reading special state information in the special state table in response to the lookup performed in the special state table;

determining from the special state information whether the next state is at least one of a start state and a terminal state;

loading current offset information into the start state register if the next state is a start state, the current offset information corresponding to the position of a character in the input character string which resulted in the next state being a start state; and

retrieving from the special state table the special state information, and retrieving the current offset information from at least one register of the plurality of start state registers when the next state is determined to be a terminal state.

7. A method for determining the start of one or more patterns of characters in an input character string, the patterns being defined by at least one character of the input character string, the input character string being provided to a system having a finite state automaton, an automaton memory operatively linked to the finite state automaton, and a plurality of start state registers operatively linked to the automaton memory and finite state automaton, the system operating in a series of states, the series of states including at least one start state and at least one terminal state, the method comprising the steps of:

providing each character of the input character string to the system such that the finite state automaton is responsive thereto and selectively transitions from a current state to a next state in response to each character;

storing in the automaton memory a state transition table and a special state table, the special state table including special state information, the special state information including start state entries and terminal state entries, the special state information having at least a first code to indicate whether the special state information is a start state entry or a terminal state entry, each start state entry including a start state register select code, each terminal state entry including a second code identifying the one or more patterns, and a start state register number code, the state transition table including current state information corresponding to the current state of the finite state automaton, character information corresponding to the characters in the input character string, next state information relating to the next state to which the finite state automaton will transition in response to the current state information and the character information, and special state table information corresponding to the next state information and indicating whether the system should perform a lookup in the special state table;

storing in each register of the plurality of start state registers information relating to the location in the input character string of the start of a particular pattern of the one or more patterns;

determining from the state transition table whether the next state is a special state in response to an input character of the input character string;

performing a lookup in the special state table if the next state is determined to be a special state;

reading at least one of the start state entries and the terminal state entries in response to the lookup performed in the special state table;

determining from the at least one of the start state entries and the terminal state entries whether the next state is at least one of a start state and a terminal state;

loading current offset information into the start state register if the next state is a start state, the current offset information corresponding to the position of a character in the input character string which resulted in the next state being a start state; and

retrieving from the special state table the terminal state entry, and retrieving the current offset information from at least one register of the plurality of start state registers when the next state is determined to be a terminal state.

8. A method for determining the start states of each rule of a plurality of rules and generating a multi-rule deterministic finite state automaton (DFA), which comprises the steps of:

prepending to each rule the metacharacters ".*" and transforming each rule prepended with the metacharacters to a non-deterministic finite state automaton (NFA) using a Thompson Construction ;

analyzing the NFA for each rule to determine NFA start states by the following substeps:

a) producing an epsilon closure starting at the initial state of the NFA;

b) producing a 1-closure of the initial epsilon closure;

c) comparing the states in the initial epsilon closure with the states in the 1-closure; and

d) determining as NFA start states all states in the 1-closure which are not in the initial epsilon closure;

converting the NFA for each rule into a DFA using an NFA to DFA algorithm, thereby creating a DFA state from one or more NFA states;

determining for each DFA state whether it contains an NFA start state;

for each DFA state that contains an NFA start state, determining the distance of the DFA state from the global start of the DFA for each rule;

comparing the distances of the DFA start states that contain an NFA start state from the global start state;

choosing as a DFA start state the DFA state containing an NFA start state which is closest to the global start state;

if more than one DFA state containing an NFA start state have the same closest distance to the global start state, accepting as DFA start states each of said closest DFA states having the same closest distance to the global start state;

creating a new NFA start state and inserting an epsilon transition from the new NFA start state to each of the DFA's for each rule, thereby creating a meta-NFA; and

converting the meta-NFA to a final multi-rule DFA.